ENGINEERING DESIGN METHOD

# PROBLEM:

"El transporte es una necesidad para todas las personas y existen diferentes medios de transporte, ya sea terrestre, aéreo o marítimo. Sin embargo, estos medios de transporte también contribuyen a la contaminación del medio ambiente. Consciente de este problema, la empresa de transporte "BadTrip" ha contratado a usted y a su equipo de software para desarrollar un sistema que encuentre la ruta más eficiente para reducir la cantidad de CO2 emitida en los viajes aéreos (tener en cuenta que es: distancia por hora y emisiones por hora), tanto nacionales como internacionales, teniendo en cuenta las distintas escalas. El sistema debe permitir agregar y eliminar las escalas necesarias para determinar la mejor ruta posible.

Es importante tener en cuenta que en ciertas épocas del año, la empresa puede no considerar la reducción de emisiones de CO2 como una prioridad, ya que en cada escala podrían perder mucho tiempo debido a las condiciones climáticas o a la temporada. En esos casos, el enfoque principal será encontrar la ruta con la menor cantidad de escalas para llegar a un destino determinado. Además, debido a que la empresa tiene varias aerolíneas ubicadas en diferentes lugares, la ruta puede comenzar desde diferentes puntos y puede haber vuelos de ida y vuelta con destinos variados."

"En cuanto al transporte terrestre, la empresa solo realiza viajes nacionales a diferentes ciudades. La empresa requiere que usted y su equipo puedan determinar la mejor ruta a partir de un punto de partida en las distintas sedes nacionales que posee, con el objetivo de encontrar la ruta más óptima para recorrer todas las ciudades dadas y reducir al máximo las emisiones de CO2. Se deben considerar tanto los viajes de ida como los de vuelta, así como la posibilidad de agregar o eliminar diferentes lugares. Además, se busca minimizar la cantidad de paradas, durante ciertas épocas del año ignorando de igual forma las emisiones de Co2. Su tarea será desarrollar un sistema que cumpla con estos requisitos y encuentre la solución más eficiente."

# PHASE 1: PROBLEM IDENTIFICATION.

La empresa de transporte "BadTrip" desea reducir la cantidad de CO2 emitida en sus viajes aéreos y terrestres, con el objetivo de mitigar el impacto ambiental aunque está restringido por la época del año. Además, se requiere un sistema de software que encuentre las rutas más eficientes para los viajes aéreos y terrestres, ya sean nacionales o internacionales, considerando las diferentes escalas o destinos.

Requerimientos funcionales:

R1 - Cálculo de la ruta más eficiente: El sistema debe ser capaz de calcular la ruta más eficiente para reducir la cantidad de combustible en los viajes aéreos y terrestres. Debe considerar los diferentes medios de transporte, como aviones y vehículos terrestres, y optimizar la ruta teniendo en cuenta la cantidad de escalas o ciudades a visitar.

R2 - Consideración de escalas y destinos: El sistema debe permitir indicar la cantidad de escalas para los vuelos, ya sean nacionales o internacionales, considerando también diferentes destinos. Debe ser capaz de determinar la mejor ruta considerando las escalas y los vuelos de ida y vuelta con destinos variados.

R3 - Optimización de emisiones de CO2: El sistema debe tener en cuenta la reducción de emisiones de CO2 como uno de los criterios principales para determinar la mejor ruta. Debe considerar la distancia, el tipo de medio de transporte utilizado y otros factores relevantes para minimizar las emisiones contaminantes.

R4- Rutas terrestres nacionales: Para los viajes terrestres nacionales, el sistema debe ser capaz de determinar la mejor ruta a partir de un punto de partida en las distintas sedes nacionales de la empresa. Debe encontrar la ruta más corta para recorrer todas las ciudades dadas y reducir al máximo las emisiones de CO2, teniendo en cuenta los viajes de ida y vuelta.

R5- Interfaz de usuario intuitiva: El sistema debe contar con una interfaz de usuario intuitiva que permita ingresar los datos necesarios del tipo de viaje, como puntos de partida, transporte aéreo o terrestre, destinos, cantidad de escalas, etc. Debe proporcionar resultados claros y fáciles de entender, mostrando la ruta óptima y la cantidad estimada de emisiones de CO2 reducidas.

R6-agregar parada o escala: El sistema debe permitir la adición o eliminación de paradas o escalas durante el trayecto en tiempo real. Los usuarios (operadores de vuelo o conductores) deben tener la capacidad de tomar decisiones sobre agregar o quitar paradas o escalas según las condiciones del viaje.

R7-Rutas mas optima: El sistema debe seleccionar las rutas mas optimas para llegar de un punto a otro bien sea por via aérea o terrestre.

# PHASE 2: COLLECTION OF THE NECESSARY INFORMATION.

En la fase de diseño, se busca definir la arquitectura y los componentes principales del sistema de optimización de rutas para reducir el impacto ambiental del transporte público. A continuación, se detallan los aspectos clave de cada requerimiento funcional y se presentan posibles soluciones:

R1. Recopilación de datos geográficos:

- Solución propuesta: Utilizar tecnologías de geolocalización y mapeo para obtener información detallada sobre la ubicación de las paradas de transporte público, las vías de conexión y otros elementos relevantes. Se pueden utilizar servicios de mapas en línea o bases de datos geoespaciales para almacenar y acceder a estos datos.

R2. Recopilación de datos de demanda:

- Solución propuesta: Implementar un sistema de registro de datos en cada parada de transporte público para capturar la demanda de pasajeros. Esto puede incluir la instalación de sensores de conteo de pasajeros y la integración de sistemas de pago electrónico que registren el origen y destino de los usuarios.

R3. Recopilación de datos de consumo de combustible:

- Solución propuesta: Establecer un proceso de recolección de datos de consumo de combustible de los vehículos de transporte público utilizados en la zona. Esto puede implicar la colaboración con las empresas de transporte público para recopilar información sobre el consumo de combustible de sus flotas y mantener una base de datos actualizada.

R4. Algoritmos de optimización de grafos:

- Solución propuesta: Implementar algoritmos de optimización de grafos adaptados al problema específico de encontrar las rutas óptimas que minimicen el tiempo de recorrido y la distancia total. Se pueden utilizar algoritmos como el algoritmo de Dijkstra o el algoritmo A\*, teniendo en cuenta las restricciones y consideraciones específicas del transporte público.

R5. Software o plataforma de cálculo y simulación:

- Solución propuesta: Desarrollar un software especializado que integre los datos geográficos, de demanda y de consumo de combustible. Esta plataforma debe permitir el cálculo y la simulación de las rutas óptimas utilizando los algoritmos de optimización de grafos. Además, debe contar con una interfaz intuitiva para que los usuarios puedan ingresar y visualizar datos, así como configurar parámetros de optimización.

R6-agregar paradas o escala:

-Soucion propuesta: El sistema debe contar con la posibilidad de agregar o quitar paradas

R7- Rutas más optimas:

-Solucion Propuesta: Algoritmos de optimización de rutas: El sistema debe utilizar algoritmos de optimización de rutas que tengan en cuenta diversos factores, como la distancia, el tiempo de viaje, las emisiones de CO2 y las condiciones específicas del viaje. Estos algoritmos analizarán múltiples opciones de rutas y seleccionarán la más eficiente en función de los criterios establecidos.

# PHASE 3: SEARCH FOR CREATIVE SOLUTIONS.

Se desarrolla el software o la plataforma de cálculo y simulación necesarios para procesar los datos geográficos, de demanda y de consumo de combustible. Este software debe utilizar algoritmos de optimización de grafos adaptados para el problema específico, como el algoritmo de Dijkstra o el algoritmo A\*.

Los datos geográficos, de demanda y de consumo de combustible se integran en el software. Esto implica recopilar información precisa sobre la ubicación de las paradas de transporte público, las vías de conexión, los límites de velocidad, la demanda de pasajeros en cada parada y los datos de consumo de combustible de los vehículos.

Se configuran los algoritmos de optimización para que tengan en cuenta tanto el tiempo de recorrido como la distancia total, con el objetivo de minimizar el consumo de combustible y las emisiones de gases contaminantes.

Utilizando el software desarrollado y los algoritmos configurados, se realizan simulaciones y cálculos para encontrar las rutas óptimas para los vehículos de transporte público. Estas rutas deben minimizar tanto el tiempo de recorrido como la distancia total, teniendo en cuenta la demanda de pasajeros y el consumo de combustible.

Se evalúan los resultados obtenidos comparando las rutas optimizadas con las rutas existentes utilizadas por los vehículos de transporte público. Se analiza el tiempo de recorrido, la distancia total y el consumo de combustible. Se pueden realizar pruebas piloto y análisis para determinar la efectividad de las nuevas rutas optimizadas en la reducción del impacto ambiental.

Según los resultados y la evaluación, se realizan ajustes y mejoras en el software, los algoritmos o los datos utilizados. Esto puede incluir correcciones de errores, optimización de algoritmos o actualización de los datos recopilados. El ciclo de implementación y evaluación puede repetirse varias veces hasta lograr una solución óptima y eficiente.

# PHASE 4: TRANSITION FROM IDEAS FORMULATION TO PRELIMINARY DESIGNS.

Se realizan pruebas utilizando datos geográficos, de demanda y de consumo de combustible para asegurar que las rutas óptimas se generen de manera precisa y confiable. Implementación gradual: Dependiendo de las circunstancias y recursos disponibles, se puede optar por implementar la solución de manera gradual. Esto implica seleccionar una zona específica o una línea de transporte público para comenzar a utilizar las rutas optimizadas. A medida que se demuestre la efectividad y los beneficios de la solución, se puede ampliar su implementación a otras áreas o líneas de transporte. Se brinda formación sobre el funcionamiento del software, la interpretación de los resultados y los ajustes que se pueden realizar. Esto asegura que el personal esté familiarizado con la solución y pueda utilizarla de manera efectiva en su trabajo diario. Se supervisa el rendimiento de las rutas optimizadas, se recopilan datos en tiempo real y se realizan análisis para identificar posibles mejoras y ajustes. Además, se brinda soporte técnico para solucionar cualquier problema que pueda surgir durante el uso diario de la solución. Se analiza el impacto ambiental, la eficiencia del transporte público y la satisfacción de los usuarios para evaluar el desempeño y los beneficios de la solución implementada. Estos resultados ayudan a retroalimentar el sistema y realizar ajustes adicionales si es necesario, con el objetivo de mantener y mejorar la efectividad de la solución en el tiempo.

# PHASE 5: EVALUATION AND SELECTION OF THE BEST SOLUTION

Durante la fase de evaluación y selección de la mejor solución, se identificaron las siguientes opciones como las más efectivas:

1. Rutas óptimas basadas en datos: Se encontró que las rutas generadas utilizando datos geográficos, de demanda y de consumo de combustible son altamente precisas y confiables. Estas rutas optimizadas han demostrado reducir los tiempos de viaje y minimizar la distancia recorrida, lo que resulta en una mayor eficiencia del transporte público.
2. Implementación gradual: La opción de implementar la solución de manera gradual ha demostrado ser efectiva. Comenzar por una zona específica o una línea de transporte público permite probar y demostrar la eficacia de las rutas optimizadas en un entorno controlado. A medida que se observan resultados positivos, se puede ampliar la implementación a otras áreas o líneas de transporte.
3. Formación y capacitación: Brindar formación adecuada sobre el funcionamiento del software y la interpretación de los resultados ha sido una opción exitosa. El personal capacitado se familiariza con la solución y puede utilizarla de manera efectiva en su trabajo diario, lo que contribuye a maximizar los beneficios obtenidos.
4. Monitoreo y análisis continuo: El monitoreo constante del rendimiento de las rutas optimizadas, la recopilación de datos en tiempo real y el análisis sistemático han sido fundamentales para identificar posibles mejoras y ajustes. Este enfoque permite mantener la efectividad de la solución a lo largo del tiempo y adaptarse a los cambios en las condiciones del tráfico y la demanda.

Estas opciones han sido identificadas como las mejores durante la fase de evaluación y selección debido a su impacto positivo en la eficiencia del transporte público y en la experiencia de los usuarios. Al considerar estas opciones, se asegura que la solución implementada sea óptima y se ajuste a las necesidades y objetivos del sistema de transporte público.

**TABLAS TAD**

|  |
| --- |
| TAD<Dijkstra> |
| |  | | --- | | Dijkstra={ index <T>} | |
| { inv: ingresar un numero positivo de vertices} |
| addVertex(T vertex)  addEdge(T source, T destination, int weight)  dijkstra(T start)  Dijkstra (constructor) |

|  |
| --- |
| addVertex(T vertex)  { Agrega un nuevo vértice al grafo.}  Pre:{ No tiene ningún pre-requisito.}  Post:{ Se agrega un nuevo vértice al grafo representado por adjacencyMap. Se crea una entrada en adjacencyMap con el vértice dado y se le asocia un mapa vacío de vecinos.} |
|  |

|  |
| --- |
| addEdge(T source, T destination, int weight)  { Agrega una arista al grafo con un peso dado.}  Pre:{ Los vértices source y destination deben existir en el grafo.}  Post:{ Se agrega una arista al grafo con el peso dado. Se actualiza el mapa de vecinos de los vértices source y destination en adjacencyMap para incluir la conexión entre ellos con el peso correspondiente.} |

|  |
| --- |
| dijkstra(T start)  { Realiza el algoritmo de Dijkstra para encontrar las distancias más cortas desde un vértice de inicio.}  Pre:{ El vértice de inicio start debe existir en el grafo.}  Post:{ Se ejecuta el algoritmo de Dijkstra en el grafo para encontrar las distancias más cortas desde el vértice de inicio. Se devuelve un mapa que asigna cada vértice a su distancia más corta desde el vértice de inicio.} |

|  |
| --- |
| Dijkstra()  {constructor de Dijkstra }  Pre:{}  Post:{constructor} |

|  |
| --- |
| TAD<BFS > |
| |  | | --- | | BFS={ index <T>} | |
| { inv: una estructura de datos que representa el grafo en el que se realizará la búsqueda.} |
| bfs(start, destination):  reconstructPath(start, destination, parentMap):  getShortestPathDj(start, destination):  printGraph() |

|  |
| --- |
| bfs(start, destination):  { realiza la búsqueda BFS desde el vértice de inicio hasta el vértice de destino.}  Pre:{ El grafo (adjacencyMap) debe estar inicializado y contener las conexiones entre los vértices.}  Post:{ Devuelve una lista de vértices que representan el camino encontrado desde el inicio hasta el destino (si existe).} |
|  |

|  |
| --- |
| reconstructPath(start, destination, parentMap):  { reconstruye el camino desde el vértice de inicio hasta el vértice de destino utilizando el parentMap.}  Pre El mapa de padres (parentMap) debe contener los padres de cada vértice en el camino.}  Post:{ El vértice de inicio (start) y el vértice de destino (destination) deben existir en el parentMap.} |

|  |
| --- |
| getShortestPathDj(start, destination):  { obtiene el camino más corto utilizando el algoritmo de Dijkstra.}  Pre:{El grafo debe estar correctamente inicializado y contener los vértices y conexiones adecuadas.}  Post:{El vértice de inicio (start) y el vértice de destino (destination) deben existir en el grafo.} |

|  |
| --- |
| printGraph():  { imprime el grafo en la consola mostrando los vértices y sus conexiones.}  Pre:{El grafo debe estar correctamente inicializado y contener los vértices y conexiones adecuadas. }  Post:{Se imprime en la consola el grafo, mostrando los vértices y sus conexiones.} |